1. What is Power BI and how does it differ from Excel?

Ans: Power BI (Power Business Intelligence) is Microsoft’s tool which is used for Data Visualization, using power BI we can extract data from various sources like SQL, Python, Azure cloud etc. Using Power Query editor, we can perform Data Cleaning and transformation. We can also defined relationship between various data sets,

We can create various charts(visuals) ex. Bar chart, KPI, line Chart, scatter chart etc. using power BI.

Three different applications are available for Power BI

* **Power BI Desktop** – A Windows desktop application for creating reports.
* **Power BI Service** – An online SaaS (Software as a Service) platform for sharing and collaborating.
* **Power BI Mobile** – Mobile apps to access reports on the go.
* **Power BI Gateway** – Connects on-premises data to the Power BI service.

We can do all the above-mentioned things using Excel as well. But there are some differences between them.

DIFFERENCE:

1. Chart created using Power BI is more advance and interactive then Excel Charts
2. We can create Power BI charts with in less time than Excel Chart.
3. We can publish reports in Power BI services, Excel does not provide this kind of service.
4. Explain the concept of data modelling in Power BI.

**Data modeling** in Power BI refers to the process of structuring your data from various sources into a logical, organized, and efficient format for analysis and visualization. It involves creating **relationships between tables**, defining **calculated columns or measures**, and organizing your data to support meaningful insights.

**Key Components of Data Modellings in Power BI**

1. **Tables**  
   These are the core building blocks. You load data into Power BI in the form of tables (from Excel, SQL, CSV, etc.).
2. **Relationships**  
   Like in a database, relationships are defined between tables using **primary and foreign keys**. Power BI uses these to link your data.
   * **One-to-many** (most common)
   * **Many-to-one**
   * **Many-to-many**
3. **Star Schema vs. Snowflake Schema**
   * **Star Schema**: A central fact table connected to dimension tables.
   * **Snowflake Schema**: Dimension tables are normalized into sub-dimensions.
4. **Calculated Columns**  
   New columns created using DAX (Data Analysis Expressions) for row-level operations.
5. **Measures**  
   Aggregated values (e.g., sum of sales, average, count) also created using DAX but calculated dynamically.
6. **Hierarchies**  
   Used to drill down into data (e.g., Year > Quarter > Month).
7. **Data Types & Formatting**  
   Correct data types are crucial for calculations, relationships, and visuals to work correctly.
8. What are the different types of connections available in Power BI?

In Power BI, there are **three main types of data connections** you can use to bring in data from various sources. Each has its own purpose and trade-offs in terms of performance, flexibility, and use cases.

| **Feature** | **Import(In memory)** | **DirectQuery** | **Live Connection** |
| --- | --- | --- | --- |
| Data Location | Stored in Power BI | Remains in source | Remains in source |
| Performance | Fastest | Depends on source | Depends on model |
| Data Size | Limited to memory | No size limit | No size limit |
| Real-Time | No | Yes | Yes |
| DAX & Modeling | Full | Limited | Very limited |
| Use Case | High-speed reports | Live dashboards | Centralized data modeling |

1. How do you handle data transformation in Power BI?

In **Power BI**, data transformation is primarily handled using the **Power Query Editor**, a built-in tool that allows you to clean, shape, and prepare your data before loading it into the data model for analysis and visualization.

**Open Power Query Editor**

* Go to **Home** → click **Transform Data**
* This launches the **Power Query Editor**, where you can apply transformation steps

**Common Data Transformation Tasks**

| **Transformation** | **Description** |
| --- | --- |
| **Remove Columns/Rows** | Delete unnecessary data to reduce clutter |
| **Filter Rows** | Include only specific values (e.g., filter by date or region) |
| **Rename Columns** | Make field names readable and consistent |
| **Change Data Types** | Ensure correct types (text, number, date, etc.) for analysis |
| **Split/Combine Columns** | Use functions like "Split by delimiter" or "Merge Columns" |
| **Replace Values** | Clean data by replacing incorrect or missing values |
| **Remove Duplicates** | Eliminate repeated records based on one or more columns |
| **Group By** | Aggregate data (e.g., total sales by category) |
| **Pivot/Unpivot** | Reshape data for proper modeling or reporting |
| **Add Custom Columns** | Use M language or built-in functions for new calculations |
| **Append/ Merge Query** | Append and merge data of different tables. |

**After Transformation**

Once your data is transformed:

* Click **Close & Apply** to load it into Power BI's data model
* Then you can start building relationships, DAX measures, and reports

5) What is DAX (Data Analysis Expressions) and why is it important in

Power BI?

DAX (Data Analysis Expressions) is **functional formula language** which is used to perform various operation on existing columns and create new columns or measures.

**IMPORTANCE:** DAX is used to perform Data Transformation on available data set.

For ex. Suppose you have Sales Data which contains, sales\_id, order\_date, unit\_sold, unit\_price.

**Types of DAX Expressions:**

* **Calculate column**

Using following DAX formula we can create new column which contains Total amount.

* Example:

Total amount = CALCULATE(Sales[“unit\_sold”]\* Sales[“unit\_price”])

It will create new calculated column With name Total amount, each rows contain values according the formula.

* **Measures**
* Measures is used to calculate single value using aggregate function like count, sum, min ,max, average etc.
* More efficient than calculated columns
* Example:

Total Sales = SUM(Sales[Amount])

* **Calculated Tables**
* Create new tables derived from existing data
* Example:

HighSales = FILTER(Sales, Sales[Amount] > 1000)

6) Can you explain the difference between calculated columns and

measures in Power BI?

**Calculated Columns vs. Measures: Key Differences**

| **Feature** | **Calculated Column** | **Measure** |
| --- | --- | --- |
| **Definition** | Adds a new column to a table with row-by-row logic | Calculates an aggregated result based on context |
| **Storage** | Stored in the model; uses memory | Calculated on-the-fly; does not consume extra storage |
| **Evaluation Context** | Row context (one row at a time) | Filter context (depends on filters, slicers, visuals) |
| **Performance** | Can impact performance due to memory usage | More efficient and preferred for aggregations |
| **Use Case** | When you need new data fields per row | When you need summaries, totals, or KPIs |

**Examples**

**Calculated Column Example**

Suppose you want to create a full name field:

Full Name = Customers[FirstName] & " " & Customers[LastName]

* This adds a new column to the Customers table.
* It will show a value **for each row**.

**Measure Example**

Now, let’s calculate **total sales**:

Total Sales = SUM(Sales[Amount])

* This doesn't add a column.
* It returns a **single value** depending on the visual’s filters (like region, date, etc.).

**When to Use What?**

* Use a **calculated column** if:
  + You need the value available for **filtering, sorting, or grouping**.
  + You need it as a part of relationships or hierarchy.
* Use a **measure** if:
  + You need **aggregated insights** (totals, averages, percentages).
  + You want **better performance and dynamic behavior** in visuals.

7) How do you handle relationships between tables in Power BI?

Handling relationships between tables in Power BI is a **core part of data modeling**, allowing you to link different tables and analyze them together in a report. This is similar to working with relationships in SQL databases (like primary and foreign keys).

**Two ways to handle Relationships in Power BI**

**1. Automatic Detection**

When you load multiple tables, Power BI often **automatically detects relationships** based on column names and data types.

* Go to **Model View** to see the relationships.
* If needed, edit or delete them.

**2. Manual Creation**

You can manually create relationships:

* Go to **Model View** or **Manage Relationships**.
* Click **“New”** and define:
  + **Table A** and **Table B**
  + Matching columns (key fields)
  + **Cardinality** (One-to-many, many-to-one, etc.)
  + **Cross filter direction** (Single or Both)

**Types of Relationships**

| **Type** | **Description** |
| --- | --- |
| **One-to-Many (1:\*)** | One row in the primary table relates to many in the secondary table (most common) |
| **Many-to-One (\*:1)** | Reverse of above, also valid |
| **Many-to-Many (*:*)** | Used when both tables have duplicates (less common, needs care) |
| **One-to-One (1:1)** | Each row in one table matches exactly one in the other |

**Cross Filter Direction**

| **Direction** | **Use Case** |
| --- | --- |
| **Single** | Data flows one way (default and safer) |
| **Both** | Allows filtering both ways, useful in certain scenarios like bridging two fact tables through a dimension table |

**Example Scenario**

You have:

* A Sales table with CustomerID
* A Customers table with CustomerID and Region

Create a **one-to-many** relationship from:

* Customers[CustomerID] (one)
* to Sales[CustomerID] (many)

8) What is the purpose of a Power BI Gateway?

A **Power BI Gateway** is like a **secure bridge** between your computer (or your company’s internal data) and the **Power BI service (cloud)**.

**Why do we need it?**

Sometimes your data is stored **on your local computer or a company server** (not online). Power BI in the cloud can’t reach it directly. So, you install the gateway on your computer or server, and it helps **Power BI securely access your data** when needed.

**What does it do?**

* **Connects local data to Power BI online**
* **Updates your dashboards with the latest data**
* **Keeps your data secure – it doesn’t move everything online, just sends the needed results**

**Example:**

Let’s say:

* You have an Excel file on your office computer.
* You create a Power BI report using that file.
* You publish the report to Power BI online.

Now, Power BI can’t see your Excel file on your office computer...  
**But** if you install the **gateway**, Power BI can ask the gateway to go check the Excel file, grab fresh data, and update your report — **automatically**!

**In Short: Power BI Gateway** helps Power BI service(online) talk to your Local data(computer’s data) — securely and automatically.

9) How can you schedule data refresh in Power BI Service?

When you publish a Power BI report using data from Excel, SQL, or another source, the data in the report doesn’t update automatically. **Scheduling a refresh** tells Power BI to go back to your data source and get the latest information — **on a regular schedule** (like every day or every hour).

**How to Schedule a Data Refresh (Step-by-Step)**

1. **Publish your report to Power BI Service**
   * Use Power BI Desktop → Click **Publish** → Choose a workspace
2. **Go to Power BI Service**
   * Visit https://app.powerbi.com and log in
3. **Open your Workspace**
   * Go to the workspace where you published your report
4. **Click on the Dataset (not the report)**
   * Find the dataset linked to your report
5. **Click the “...” (More Options) > Settings**
6. **Under “Scheduled Refresh”**
   * Turn it **on**
   * Set the **time**, **timezone**, and **frequency** (daily, weekly, etc.)
7. **(If Needed) Add Gateway Credentials**
   * If your data is stored on your computer or local server, make sure you’ve installed a **gateway** and entered your login details for the data source.
8. **Click Save**

**Example:**

You can schedule your sales report to refresh **every morning at 7 AM**, so when your team opens it, they see the **latest numbers** — no manual work needed.

10) Explain the concept of row-level security in Power BI.

**Row-Level Security** means **showing different data to different users** — all from the same report.

It helps you **control what each person can see** in a Power BI report based on their role, department, or other filters.

**Why Use RLS?**

Let’s say you have a sales report that includes data for:

* North region
* South region
* East region
* West region

But:

* You want the **North manager** to see only **North region data**
* The **South manager** should only see **South region data**, and so on

With **RLS**, you don’t need to create 4 reports. Just one report, and Power BI will **automatically filter the data** for each user.

**How Does RLS Work?**

1. **Define roles and rules in Power BI Desktop**
   * Example: Create a role called NorthManager  
     Rule: Region = "North"
2. **Publish the report to Power BI Service**
3. **Assign users to roles**
   * In Power BI Service, go to the dataset settings
   * Assign users or groups (like john@example.com) to the NorthManager role

Now, when John opens the report, he only sees **North region data**!

**RLS = Show only the rows of data a person is allowed to see, based on rules you set.**

It's perfect for **security**, **simplicity**, and **scalability**.

11) What is the Power BI Desktop and how does it differ from Power

BI Service?

Power BI Desktop is application which work on local machine, while Power BI service is application which work online.

We can create and design report in Power BI Desktop and publish it on Power BI Service.

Power BI Service is used to share, view and manage reports online.

| **Feature** | **Power BI Desktop** | **Power BI Service** |
| --- | --- | --- |
| **Platform** | Windows desktop app | Web-based (cloud) |
| **Purpose** | Create and design reports | Share, view, and manage reports |
| **Data Modeling** | Yes (full features) | No (view only) |
| **Data Refresh** | Manual | Scheduled / automatic |
| **Sharing** | Can’t share | Can share with others |
| **Row-Level Security Setup** | Create roles | Assign users to roles |
| **Usage** | Analysts / developers | Report viewers / decision makers |

12) Explain the concept of Direct Query in Power BI.

There are two types of connection mode

* 1. **Import mode:** which keep the copy of data set in memory.
  2. **Direct Query:** which connect with live data.

**DirectQuery** means Power BI does **not copy** your data into the report. Instead, it **connects directly to the source** (like SQL Server, Oracle, or Azure) and gets the data **live** every time you view the report.

**Key Points:**

* **Always up to date** – no need to refresh manually
* **Good for large databases**
* **No data stored in the Power BI file**
* May be **slower** depending on the data source speed

**Example**: If your data is in a live SQL database, Power BI will ask for the latest numbers every time you view the report.

13) What are Power BI templates and how are they useful?

A **Power BI template** (.PBIT file) is like a **ready-made report without data**. It includes:

* Report design
* Visuals
* DAX formulas
* Data model
* Queries

But the **data itself is not saved**, so others can use it with their own data.

**Uses:**

* Saves time — reuse the same layout for different projects
* Share reports without sharing sensitive data
* Makes it easy to **standardize** reports across teams

**Example**: You build a sales dashboard layout for one region. You export it as a **template**, and other regions can use it by just plugging in their own data.

14) How do you handle incremental data refresh in Power BI?

I**ncremental refresh** means **only new or changed data is loaded**, instead of reloading **all the data** every time — which saves time and improves performance.

**Example**

Imagine you have 5 years of sales data. You don’t need to reload all 5 years daily — just **add yesterday’s new sales**.

**Steps to set it up:**

1. **Add a Date/Time column** to your data (like "OrderDate").
2. In **Power BI Desktop**:
   * Go to **Power Query Editor**
   * Use **RangeStart** and **RangeEnd** parameters to define the date range
3. Apply **filters** using these parameters.
4. Publish to **Power BI Service**.
5. In the Service, go to the dataset settings and **enable Incremental Refresh**.
6. Set:
   * How much history to keep (e.g., last 5 years)
   * How much data to refresh (e.g., last 1 day)

After setup, only recent data gets updated during refresh.

15) What is the role of Power Query in Power BI?

**Power Query** is the **tool inside Power BI** used for **loading, cleaning, and shaping your data** before building reports.

**For Example**

A **kitchen** where you wash, cut, and prepare the ingredients (your data) before cooking the final dish (your report).

**Power Query can do following things:**

* Remove or rename columns
* Filter rows
* Merge tables
* Change data types (text, date, number, etc.)
* Add calculated columns
* Handle missing or messy data

You don’t need to write code — just click and apply steps. Power BI will remember each step as a recipe (shown in the “Applied Steps” pane).

16) Explain the difference between calculated columns and calculated tables in Power BI.

| **Calculated Column** | **Calculated Table** |
| --- | --- |
| A new **column** added to an existing table | A completely **new table** created using DAX |
| Inside existing tables (like adding "Full Name" = First + Last) | Used to create summaries, filters, or custom logic |
| Filtering, grouping, or displaying extra fields in visuals | Creating slicers, joining data, or building custom models |
| FullName = [FirstName] & " " & [LastName] | TopProducts = FILTER(Products, Products[Sales] > 1000) |

**In short**:

* Use **calculated columns** when you need a new field **inside a table**.
* Use **calculated tables** when you want to build a **new table** based on logic or filtering.

17) How do you create custom visuals in Power BI?

There are **two ways**:  
🔹 Use visuals created by others (easiest)  
🔹 Create your own (more advanced)

**A) Using Prebuilt Custom Visuals**

1. Open your report in **Power BI Desktop**
2. Go to the **Visualizations pane**
3. Click the **“... More visuals”** (3 dots)
4. Choose **“Get more visuals”**
5. Browse or search the visuals you want (like bullet chart, KPI card, etc.)
6. Click **Add** to use it in your report

These are from **Microsoft AppSource**, safe to use, and no coding needed.

**B) Creating Your Own Custom Visual (Advanced)**

1. You need to know **TypeScript** and **Power BI Developer Tools**
2. Install **Node.js** and **Power BI Visuals SDK**
3. Use the command line to create a visual project
4. Build and test your visual
5. Package it as .pbiviz
6. Import it into Power BI Desktop

Ideal for developers who want very specific or branded visuals.

18) What are the best practices for optimizing performance in

Power BI?

To make your Power BI reports **faster and smoother**, follow these **best practices**:

**Data Model Optimization**

* **Remove unused columns & tables** – Smaller models load faster
* Use **star schema** (facts and dimensions) instead of flat tables
* Prefer **numeric columns** over text for filtering and grouping

**DAX Optimization**

* Avoid complex calculations in visuals — move them to **measures**
* Use **CALCULATE** wisely, and avoid repeated use of **FILTER** inside DAX
* Avoid using **columns** when a **measure** is enough

**Data Load**

* Use **Import mode** when possible — it’s faster than DirectQuery
* For **DirectQuery**, keep visuals and data volume minimal
* Use **incremental refresh** for large datasets

**Visuals**

* Limit visuals per page (ideally **<10**) — too many slow things down
* Avoid using high-cardinality slicers (like "Customer Name" with 50,000 rows)
* Use **aggregated data** when possible

**General Tips**

* Keep **Power Query steps** simple
* Use **summary tables** or **aggregated views** when loading data
* Turn off **Auto Date/Time** for new files (File > Options > Data Load)

19) How can you integrate Power BI with other Microsoft products like Azure and Office 365?

**How to Connect with Azure:**

1. **Open Power BI Desktop**
2. Click **Home > Get Data**
3. Choose your Azure service (e.g., “Azure SQL Database”)
4. Enter connection details (server, database, credentials)
5. Choose **Import** or **DirectQuery**
6. Load your data and build visuals

Example: Connect Power BI to an **Azure SQL Database** to create live dashboards using real-time sales data.

**How to connect with office 365:**

**A) In Microsoft Teams**

1. Open your **Teams** channel
2. Click **+ (Add a tab)** > Select **Power BI**
3. Choose the report you want to share

**B) In SharePoint**

1. Copy the **report URL** from Power BI Service
2. Add a **Power BI Web Part** to your SharePoint page
3. Paste the link to show the live report

**C) In Excel**

* Use **“Analyze in Excel”** feature to explore Power BI datasets in Excel
* Or, import Excel files stored in **OneDrive** directly into Power BI

| **Integration Area** | **What You Get** |
| --- | --- |
| **Azure** | Live data connections, big data support |
| **Office 365** | Sharing, collaboration, embedded reports |

20) Explain the concept of aggregations in Power BI.

**Aggregations** in Power BI are ways to **summarize large amounts of data** so reports load faster and run smoother.

Example: Instead of loading millions of rows (like every single sale), Power BI stores **summarized data** (like total sales per month), which is much faster to work with.

**Benefits of Using Aggregations:**

* **Faster performance**
* **Less memory usage**
* Great for **large datasets** (millions or billions of rows)

**How to Use Aggregations in Power BI:**

1. Create an **aggregated table** in Power BI (manually or in Power Query)
2. Define it as an **aggregation table** in **Model view**
3. Tell Power BI which columns summarize which detailed data (like Sales = SUM(SalesAmount))
4. Power BI will **automatically use the aggregation table** if it matches a visual’s need

If a report doesn’t need full detail, Power BI uses the faster summary table.

**In short:**

* **Aggregation = Summary of data (like totals, averages)**
* Used to make reports **faster**, especially with **huge datasets**
* Power BI automatically chooses when to use them, if set up correctly

21) How do you handle error handling and data quality in Power BI?

In Power BI, ensuring **clean and accurate data** is critical before building reports. You handle **errors and data quality** mostly in **Power Query**.

**How to Handle Errors:**

**In Power Query Editor:**

1. **Remove Errors:**
   * Right-click on a column → *Remove Errors*
   * Useful if bad rows are few and not needed
2. **Replace Errors:**
   * Use *Replace Errors* to swap errors with default values like 0 or "Unknown"
3. **Use Conditional Columns:**
   * Create logic to fix or flag bad data  
     Example: if [Sales] = null then 0 else [Sales]
4. **Detect Data Types:**
   * Make sure columns have correct data types (text, number, date)
5. **Keep/Remove Rows:**
   * Filter out blank, duplicate, or invalid rows

**Good Practices for Data Quality:**

* Use **Column Profiling** (in Power Query) to find missing values, outliers
* Validate using **Data view** in Power BI Desktop
* Create **Quality Rules** (like "no nulls in primary fields", "date must be after 2000")
* Log or flag errors for review

Power BI doesn’t fix bad data by default — **you decide** how to handle it during import.

22) What is the purpose of Power BI Embedded and when would you use it?

**Power BI Embedded** is a **Microsoft Azure service** that lets developers **embed Power BI reports into custom applications or websites**.

**Example:** Instead of making users go to the **Power BI portal**, you bring **the report into your own app** — like a customer portal, sales dashboard, or mobile app.

**When to Use Power BI Embedded:**

* You want to **show Power BI reports inside your web/app** to customers or external users
* You don’t want users to have **Power BI accounts**
* You want full **control of branding, layout, and user experience**
* You need **multi-tenant support** (different customers see different data)

**How It Works (Basic Flow):**

1. Build your report in **Power BI Desktop**
2. Publish to a **Power BI Workspace** in a **premium or embedded-capacity**
3. Use **Power BI REST API or JavaScript SDK** to embed it into your app
4. Use **tokens** (like Azure AD or Embed tokens) to control user access

Example: A software company shows real-time analytics dashboards inside its customer portal using Power BI Embedded — users see their own data, without logging into Power BI.